Steven M. Hoffberg

From: Steven M. Hoffberg [steve@hoffberg.org]
Sent: Thursday, November 18, 2004 12:19 PM
To: 'Nguyen, Nga'
Subject: 09/599,163 test_client.c

```
/* ------------------------------------------------------------------------
 * test client for TVS/Clickshare service
 * Copyright (c) 1995 Newshare Corporation
 * ------------------------------------------------------------------------
 */

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/time.h>

#include "tvs.h"
#include "tvs_util.h"
#include "tvs_client.h"

#define PROGRAM     "test_client"

static int delay  = 0;
static int repeat = 1000;

/* micro "sleep" function, so I can control timing */

void
_usleep(int ms)
{
  struct timeval tm;

  tm.tv_sec = (int) (ms / 1000);
  tm.tv_usec = (ms * 10);

  /* this is the coolest Unix system call! */

  select(0, (fd_set *) NULL, (fd_set *) NULL, (fd_set *) NULL, &tm);
}

int
run_test()
{
  TVS_SERVER t;
  TVS_PROFILE up, np;
```

```
TVS_TOKEN tk[1000];
char *ep;
int i, err = 0;

unsigned long h_ip = 0x83110cb2;      /* a ficticious user host */
unsigned long u_id = 0x10c0c0a0;      /* and similar ID */

fprintf(stderr, "Clickshare Test Client\n");
fprintf(stderr,
        "using inter-request delay %d (microseconds) and repeat %d (interations)\n",
        delay, repeat);

ep = getenv("CLICKSHARE_CONF");
if (!ep) {
  fprintf(stderr, "env. variable CLICKSHARE_CONF not defined\n");
  exit(1);
}
else
  fprintf(stderr, "Reading configuration from %s...\n", ep);

LogOpen("test_client", LOG_UPTO(LOG_INFO));
LogMsg(LOG_ERR, "Clickshare Test Client starting");

/* create a user profile, and set redundant information */

up = tvs_make_user_profile();
if (!up) {
  fprintf(stderr, "could not construct basic user profile\n");
  exit(1);
}

tvs_set_pmid(up, tvs_pm_id);
tvs_set_service_class(up, 0xd);
tvs_set_adv_context(up, 0x2);
tvs_set_customer_group(up, 0xc);
tvs_set_privacy1_flag(up, 1);
tvs_set_pdac_flag(up, 1);

/* make contact with the TVS server to start "session" */

fprintf(stderr, "Initializing service...\n");
if(!(t = tvs_initialize_service(ep))) {
  fprintf(stderr, "oops, botched initialization.\n");
  exit(1);
}

/* identify which TVS server is serving me */

fprintf(stderr, "Selected service: %s\n", tvs_identify_tvs_server());
fprintf(stderr, "Looping to obtain and validate %d tokens...\n", repeat);
```

```
for (i = 0; i < repeat; i++) {

  /* set a new u_id / host_ip each time */

  u_id++;
  h_ip++;

  tvs_set_userid(up, u_id);
  tvs_set_hostid(up, h_ip);

  /* obtain token for user */

  tk[i] = tvs_new_token(up);
  if (!tk[i]) {
      fprintf(stderr, "couldnt get a token on iteration %d.\n", i);
      err++;
  }

  if (repeat < 10)
   fprintf(stderr, "new token is: %s\n", (char *) tk[i]);

  _usleep(delay);

  /* have it validated */

  np = tvs_validate_token(tk[i], h_ip);
  if (np && (repeat < 10))
   tvs_show_user_profile(np);
  if (!np) {
      fprintf(stderr, "ERROR! token not validated -- iteration %d\n", i);
      err++;
  }

  if (!(i%50) && (repeat <= 1000)) fprintf(stderr,"iteration: %d\n", i);
}

if (err) {
 fprintf(stderr,"oops, appears we had some headaches in the loop.\n");
 exit(1);
}

fprintf(stderr,"OK, got %d valid tokens.\n", repeat);

/* let's throw some junk at it */

fprintf(stderr,"Testing error modes...\n\t...real token, wrong address:\n");
np = tvs_validate_token(tk[0], h_ip);
if (np)
 tvs_show_user_profile(np);
else
 printf ("no profile!\n");
```

```
  fprintf(stderr, "\t...bogus token:\n");
  tvs_validate_token("34809ajdfn8237ry0238vsdv", h_ip);
  if (np)
   tvs_show_user_profile(np);
  else
   fprintf(stderr, "no profile!\n");

  /* afterward, call back to invalidate all of them */

  fprintf(stderr, "OK, back to valid use, invalidate existing tokens...\n");

  for (i = 0; i < repeat; i++) {
    if(! tvs_invalidate_token(tk[i], 1)) {
        fprintf(stderr, "invalidation error at iteration %d\n", i);
        err++;
    }
    else {
        if (!(i % 50) && (repeat <= 1000))
         fprintf(stderr,"iteration: %d\n", i);
    }
    _usleep(delay);
 }

  if (!err)
   fprintf(stderr, "Finished invalidating tokens\n");
  else
   fprintf(stderr, "appears to have been some hassles in the final loop.\n");

  /* shut down cleanly */

  fprintf(stderr, "Attempting clean shut down...\n");

  tvs_drop_service();
  LogMsg(LOG_ERR, "Clickshare Test Client finished");
  fprintf(stderr, "done\n");
  exit(0);
}

int
main(int argc, char *argv[])
{
 extern char *optarg;
 int c;

 while ((c = getopt(argc, argv, "d:r:")) != -1) {
  switch (c) {
  case 'd':
    delay = atoi(optarg);
    break;
  case 'r':
      repeat = atoi(optarg);
      if (repeat > 1000) repeat = 1000;
      break;
    default:
      fprintf(stderr, "usage: %s [-d delay] [-r repeat]\n", PROGRAM);
      exit(0);
    }
  }

  run_test();
  exit(0);
}
```

Very truly yours,

Steven M. Hoffberg
Milde & Hoffberg, LLP
Suite 460
10 Bank Street
White Plains, NY 10606
(914) 949-3100 tel.
(914) 949-3416 fax
steve@hoffberg.org
www.hoffberg.org